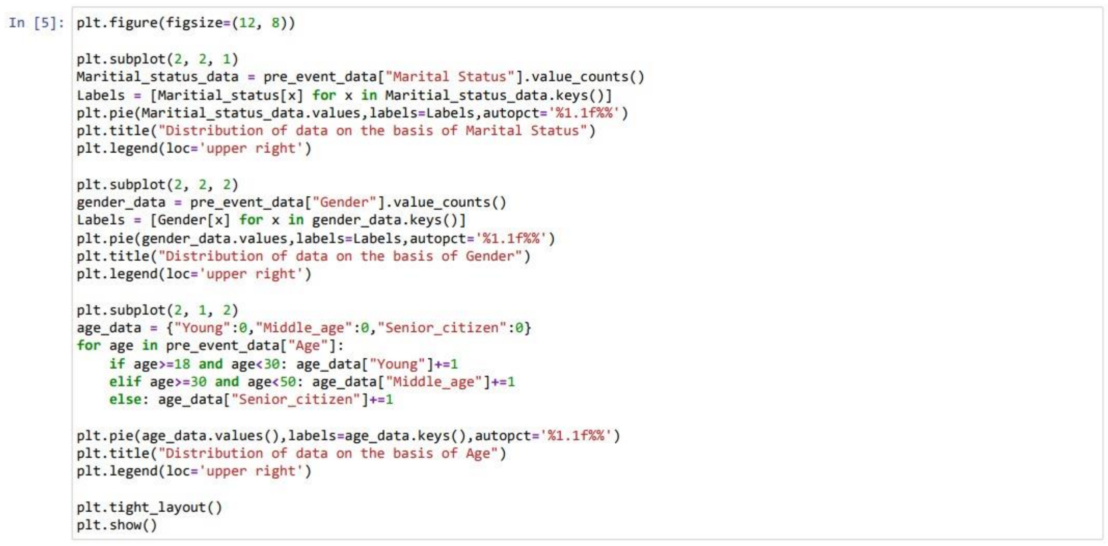
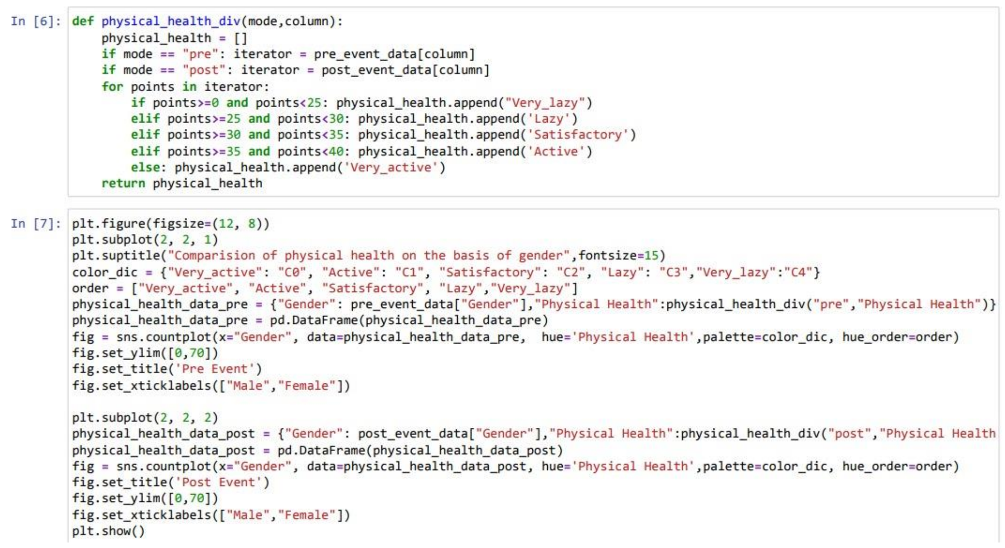


```
In [5]: plt.figure(figsize=(12, 8))

        plt.subplot(2, 2, 1)
        Maritial_status_data = pre_event_data["Marital Status"].value_counts()
        Labels = [Maritial_status[x] for x in Maritial_status_data.keys()]
        plt.pie(Maritial_status_data.values,labels=Labels,autopct='%1.1f%%')
        plt.title("Distribution of data on the basis of Marital Status")
        plt.legend(loc='upper right')

        plt.subplot(2, 2, 2)
        gender_data = pre_event_data["Gender"].value_counts()
        Labels = [Gender[x] for x in gender_data.keys()]
        plt.pie(gender_data.values,labels=Labels,autopct='%1.1f%%')
        plt.title("Distribution of data on the basis of Gender")
        plt.legend(loc='upper right')

        plt.subplot(2, 1, 2)
        age_data = {"Young":0,"Middle_age":0,"Senior_citizen":0}
        for age in pre_event_data["Age"]:
            if age>=18 and age<30: age_data["Young"]+=1
            elif age>=30 and age<50: age_data["Middle_age"]+=1
            else: age_data["Senior_citizen"]+=1

        plt.pie(age_data.values(),labels=age_data.keys(),autopct='%1.1f%%')
        plt.title("Distribution of data on the basis of Age")
        plt.legend(loc='upper right')

        plt.tight_layout()
        plt.show()
```

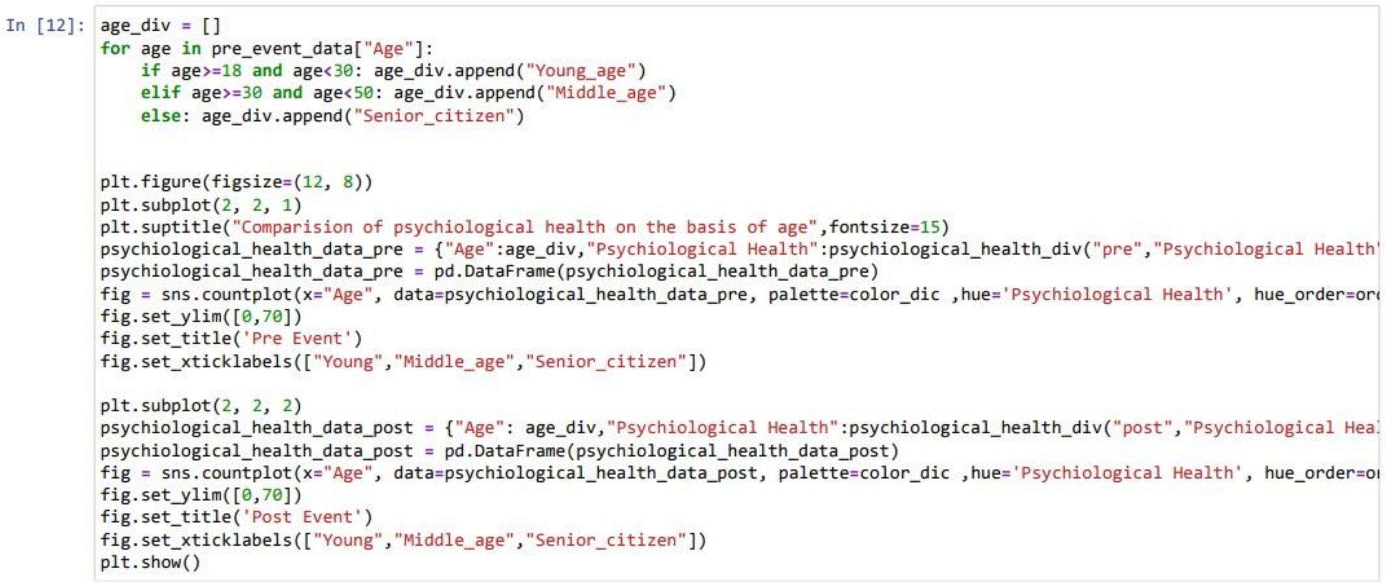
The above code generates three pie charts using Matplotlib to visualize survey data from a prisoner's meditation camp. The first subplot is a pie chart showing the distribution of prisoners based on marital status. It counts the occurrences of each marital status category, prepares the labels, and plots the data with percentages displayed. The chart is titled "Distribution of data on the basis of Marital Status" and includes a legend in the upper right corner. The second subplot presents the gender distribution of prisoners. It follows a similar process: counting the occurrences of each gender, setting up the labels, and plotting a pie chart with percentages. The chart is titled "Distribution of data on the basis of Gender" and also includes a legend in the upper right corner. The third subplot depicts the age distribution of prisoners. It categorizes the age data into "Young" (ages 18-30), "Middle_age" (ages 30-50), and "Senior_citizen" (ages 50 and above). It counts the number of prisoners in each age group, prepares the labels, and plots the pie chart with percentages. The chart is titled "Distribution of data on the basis of Age" with a legend in the upper right corner.



```
In [6]: def physical_health_div(mode,column):
            physical_health = []
            if mode == "pre": iterator = pre_event_data[column]
            if mode == "post": iterator = post_event_data[column]
            for points in iterator:
                if points>=0 and points<25: physical_health.append("Very_lazy")
                elif points>=25 and points<30: physical_health.append('Lazy')
                elif points>=30 and points<35: physical_health.append('Satisfactory')
                elif points>=35 and points<40: physical_health.append('Active')
                else: physical_health.append('Very_active')
            return physical_health
```

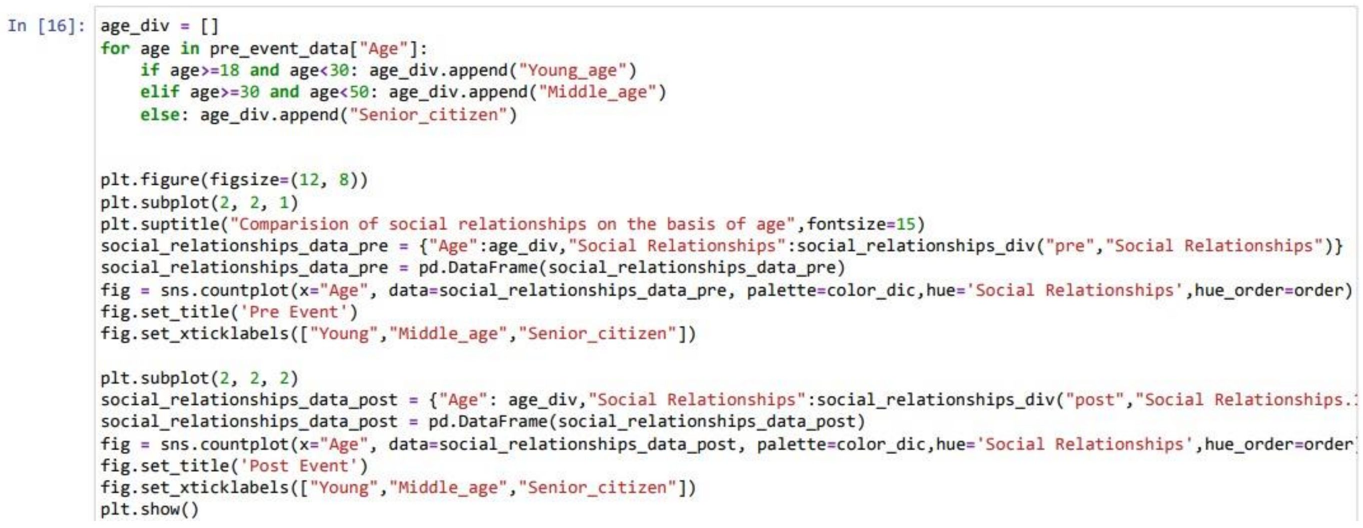
```
In [7]: plt.figure(figsize=(12, 8))
        plt.subplot(2, 2, 1)
        plt.suptitle("Comparision of physical health on the basis of gender",fontsize=15)
        color_dic = {"Very_active": "C0", "Active": "C1", "Satisfactory": "C2", "Lazy": "C3","Very_lazy":"C4"}
        order = ["Very_active", "Active", "Satisfactory", "Lazy","Very_lazy"]
        physical_health_data_pre = {"Gender": pre_event_data["Gender"],"Physical Health":physical_health_div("pre","Physical Health")}
        physical_health_data_pre = pd.DataFrame(physical_health_data_pre)
        fig = sns.countplot(x="Gender", data=physical_health_data_pre,  hue='Physical Health',palette=color_dic, hue_order=order)
        fig.set_ylim([0,70])
        fig.set_title('Pre Event')
        fig.set_xticklabels(["Male","Female"])

        plt.subplot(2, 2, 2)
        physical_health_data_post = {"Gender": post_event_data["Gender"],"Physical Health":physical_health_div("post","Physical Health
        physical_health_data_post = pd.DataFrame(physical_health_data_post)
        fig = sns.countplot(x="Gender", data=physical_health_data_post, hue='Physical Health',palette=color_dic, hue_order=order)
        fig.set_title('Post Event')
        fig.set_ylim([0,70])
        fig.set_xticklabels(["Male","Female"])
        plt.show()
```

The provided code creates visualizations to compare physical health based on gender before and after an event, using Seaborn and Matplotlib libraries. The physical_health_div function classifies physical health into categories: "Very_lazy," "Lazy," "Satisfactory," "Active," and "Very_active," based on specified point ranges. The function takes a mode ("pre" or "post") and a column name to iterate over the relevant dataset and returns a list of physical health categories. The first subplot is created to visualize the pre-event physical health distribution by gender. The color_dic dictionary defines the colour scheme for each physical health category, and order specifies the sequence of categories. The pre-event data is prepared, and a Seaborn count plot is created, with gender on the x-axis and physical health as the hue. The y-axis limit is set from 0 to 70, and the subplot is titled "Pre Event." The second subplot visualizes the post-event physical health distribution by gender, following the same process. The post-event data is prepared, and a similar Seaborn count plot is generated. This subplot is titled "Post Event." Finally, the layout is adjusted to fit all subplots neatly, and the plots are displayed using plt.show(). This code effectively compares the physical health changes before and after the event across different genders.



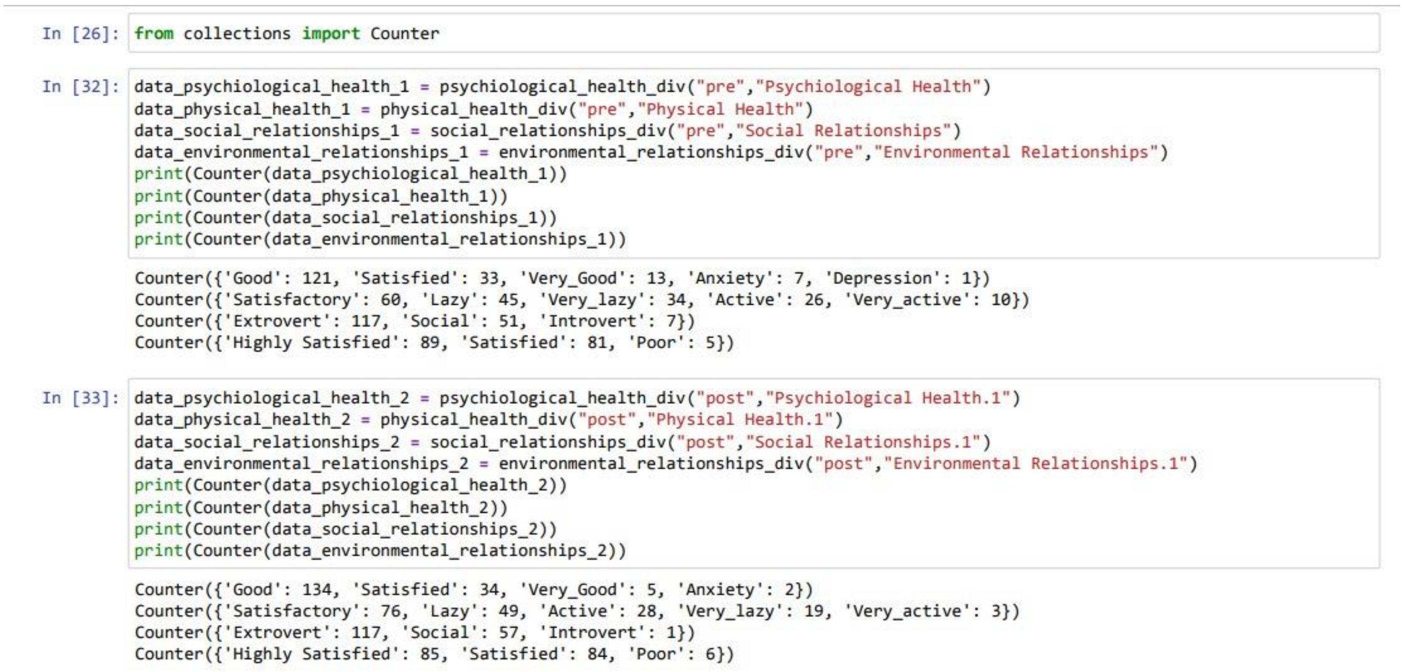
```
In [12]: age_div = []
         for age in pre_event_data["Age"]:
             if age>=18 and age<30: age_div.append("Young_age")
             elif age>=30 and age<50: age_div.append("Middle_age")
             else: age_div.append("Senior_citizen")


         plt.figure(figsize=(12, 8))
         plt.subplot(2, 2, 1)
         plt.suptitle("Comparision of psychiological health on the basis of age",fontsize=15)
         psychiological_health_data_pre = {"Age":age_div,"Psychiological Health":psychiological_health_div("pre","Psychiological Health'
         psychiological_health_data_pre = pd.DataFrame(psychiological_health_data_pre)
         fig = sns.countplot(x="Age", data=psychiological_health_data_pre, palette=color_dic ,hue='Psychiological Health', hue_order=or
         fig.set_ylim([0,70])
         fig.set_title('Pre Event')
         fig.set_xticklabels(["Young","Middle_age","Senior_citizen"])

         plt.subplot(2, 2, 2)
         psychiological_health_data_post = {"Age": age_div,"Psychiological Health":psychiological_health_div("post","Psychiological Hea
         psychiological_health_data_post = pd.DataFrame(psychiological_health_data_post)
         fig = sns.countplot(x="Age", data=psychiological_health_data_post, palette=color_dic ,hue='Psychiological Health', hue_order=o
         fig.set_ylim([0,70])
         fig.set_title('Post Event')
         fig.set_xticklabels(["Young","Middle_age","Senior_citizen"])
         plt.show()
```

This code compares the psychological health of prisoners based on their age before and after a meditation event, using Matplotlib and Seaborn libraries. The age_div list categorizes ages into "Young_age" (18-30 years), "Middle_age" (30-50 years), and "Senior_citizen" (above 50 years). A subplot grid of 2x2 is defined for the plots. The subtitle "Comparison of psychological health on the basis of age" is set with a font size of 15. The pre-event psychological health data is prepared by classifying physical health using the psychological_health_div function, similar to the earlier scripts, and combining it with age data. This data is then visualized using a Seaborn count plot, with age on the x-axis and psychological health as the hue. The y-axis limit is set from 0 to 70, and the title "Pre Event" is added. The second subplot focuses on the post-event psychological health data. It follows the same process: the post-event data is classified, prepared, and visualized using another Seaborn count plot. The y-axis limit is again set from 0 to 70, and this subplot is titled "Post Event." The x-axis labels are set to "Young," "Middle_age," and "Senior_citizen" for both subplots. Finally, plt.show() is called to display the plots. This visualization effectively shows changes in psychological health before and after the event, segmented by age groups.



```
In [16]: age_div = []
         for age in pre_event_data["Age"]:
             if age>=18 and age<30: age_div.append("Young_age")
             elif age>=30 and age<50: age_div.append("Middle_age")
             else: age_div.append("Senior_citizen")


         plt.figure(figsize=(12, 8))
         plt.subplot(2, 2, 1)
         plt.suptitle("Comparision of social relationships on the basis of age",fontsize=15)
         social_relationships_data_pre = {"Age":age_div,"Social Relationships":social_relationships_div("pre","Social Relationships")}
         social_relationships_data_pre = pd.DataFrame(social_relationships_data_pre)
         fig = sns.countplot(x="Age", data=social_relationships_data_pre, palette=color_dic,hue='Social Relationships',hue_order=order)
         fig.set_title('Pre Event')
         fig.set_xticklabels(["Young","Middle_age","Senior_citizen"])

         plt.subplot(2, 2, 2)
         social_relationships_data_post = {"Age": age_div,"Social Relationships":social_relationships_div("post","Social Relationships.
         social_relationships_data_post = pd.DataFrame(social_relationships_data_post)
         fig = sns.countplot(x="Age", data=social_relationships_data_post, palette=color_dic,hue='Social Relationships',hue_order=order
         fig.set_title('Post Event')
         fig.set_xticklabels(["Young","Middle_age","Senior_citizen"])
         plt.show()
```

The image presents a comprehensive visualization of a survey conducted to assess the impact of a meditation session on social relationships among prison inmates. The analysis categorizes participants into three age groups: young adults (18-29), middle-aged individuals (30-49), and senior citizens (50+). The visualization consists of two side-by-side bar charts, comparing social relationships across these age groups before and after the meditation event. Each chart uses color-coding to represent different types of social relationships, likely positive, neutral, and negative. This approach allows for a clear comparison of how the meditation session affected social dynamics within the prison population, taking into account age differences. The use of pre-event and post-event comparisons provides insight into the immediate effects of meditation on inmates' social perceptions and interactions. This type of analysis can be valuable for understanding the potential benefits of meditation programs in correctional facilities and how they might be tailored to different age groups for maximum effectiveness.



```
In [26]: from collections import Counter
```

```
In [32]: data_psychiological_health_1 = psychiological_health_div("pre","Psychiological Health")
         data_physical_health_1 = physical_health_div("pre","Physical Health")
         data_social_relationships_1 = social_relationships_div("pre","Social Relationships")
         data_environmental_relationships_1 = environmental_relationships_div("pre","Environmental Relationships")
         print(Counter(data_psychiological_health_1))
         print(Counter(data_physical_health_1))
         print(Counter(data_social_relationships_1))
         print(Counter(data_environmental_relationships_1))

         Counter({'Good': 121, 'Satisfied': 33, 'Very_Good': 13, 'Anxiety': 7, 'Depression': 1})
         Counter({'Satisfactory': 60, 'Lazy': 45, 'Very_lazy': 34, 'Active': 26, 'Very_active': 10})
         Counter({'Extrovert': 117, 'Social': 51, 'Introvert': 7})
         Counter({'Highly Satisfied': 89, 'Satisfied': 81, 'Poor': 5})
```

```
In [33]: data_psychiological_health_2 = psychiological_health_div("post","Psychiological Health.1")
         data_physical_health_2 = physical_health_div("post","Physical Health.1")
         data_social_relationships_2 = social_relationships_div("post","Social Relationships.1")
         data_environmental_relationships_2 = environmental_relationships_div("post","Environmental Relationships.1")
         print(Counter(data_psychiological_health_2))
         print(Counter(data_physical_health_2))
         print(Counter(data_social_relationships_2))
         print(Counter(data_environmental_relationships_2))

         Counter({'Good': 134, 'Satisfied': 34, 'Very_Good': 5, 'Anxiety': 2})
         Counter({'Satisfactory': 76, 'Lazy': 49, 'Active': 28, 'Very_lazy': 19, 'Very_active': 3})
         Counter({'Extrovert': 117, 'Social': 57, 'Introvert': 1})
         Counter({'Highly Satisfied': 85, 'Satisfied': 84, 'Poor': 6})
```

The code analyses survey data from a meditation session held in a prison, focusing on four key areas: psychological health, physical health, social relationships, and environmental relationships. The code utilizes the **Counter** class from the **collections** module to count the occurrences of different responses in each category, both before ("pre") and after ("post") the meditation session. For psychological health, the "pre" data shows 121 responses of "Good," 33 of "Satisfied," 13 of "Very_Good," 7 of "Anxiety," and 1 of "Depression." In contrast, the "post" data indicates an improvement with 134 "Good," 34 "Satisfied," 5 "Very_Good," and a reduction in negative responses to 2 "Anxiety." Physical health responses also show changes, with "pre" data including 60 "Satisfactory," 45 "Lazy," 34 "Very_lazy," 26 "Active," and 10 "Very_active," while "post" data shows 76 "Satisfactory," 49 "Lazy," 28 "Active," 19 "Very_lazy," and 3 "Very_active." Social relationships remain relatively stable, with "pre" data showing 117 "Extrovert," 51 "Social," and 7 "Introvert," and "post" data showing 117 "Extrovert," 57 "Social," and 1 "Introvert." Environmental relationships also exhibit slight changes, with "pre" data showing 89 "Highly Satisfied," 81 "Satisfied," and 5 "Poor," and "post" data showing 85 "Highly Satisfied," 84 "Satisfied," and 6 "Poor." Overall, the code highlights the impact of the meditation session on the prisoners' mental, physical, and social well-being.